OOPS using java

A MINI-PROJECT BY:

MANIKANDAN S 230701175

MOHAMED IKRAM 230701188

MITHUN AR 230701186

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project **“UG DISSERTATION”** is the bonafide work of **“MANIKANDAN S , MOHAMED IKRAM K H D,MITHUN A R”** who carried out the project work under my supervision.

Submitted for the practical examination held on _____23/11/2024_____

SIGNATURE SIGNATURE

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

UG dissertation , sometimes known as a thesis is an research project completed as a part of undergraduate or postgraduate degree. Typically, a dissertation allows students present their findings in response to a question or proposition that they choose themselves.

The project tests the independent research skills students have acquired during their time at university, with the assessment used to help determine their final grade. Although there is usually some guidance from the tutors, the dissertation project is largely independent.

Our project aims to streamline UG dissertation process and provide support to students. Students will register on the website , decide what research topics interest them and submit it on the website.

The dissertation will be evaluated and the results will be posited on the website.
Students will be able to store their data and review them when required on the website.

TABLE OF CONTENTS

INTRODUCTION

INTRODUCTION

The Railway Ticket Booking and Management System is a comprehensive software application designed to facilitate the smooth and efficient operation of train ticket reservations, cancellations, and management of train schedules. This project integrates a user-friendly interface with a backend database to handle various operations like booking tickets, searching for available trains, managing user profiles, and handling administrative tasks like updating train schedules and managing seat availability.

Developed using Java for its versatility and ease of integration with databases, and MySQL as the relational database management system for secure and scalable data storage, this system aims to streamline the entire ticketing process. Whether it’s a passenger searching for trains or an administrator managing routes and schedules, this system provides an intuitive interface for all users. The project utilizes Java's object-oriented capabilities for code reusability and scalability, while MySQL ensures efficient data storage, retrieval, and manipulation.

The objective of this project is to reduce manual intervention, enhance operational efficiency, and provide a seamless booking experience to users, while also enabling easy management for railway authorities.

IMPLEMENTATION

The **UG DISSERTATION** project discussed here is implemented using the concepts of **JAVA SWINGS** and **MYSQL**.

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS:

PROCESSOR : Intel i5

MEMORY SIZE : 4GB(Minimum)

HARD DISK : 500 GB of free space

2.2 SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE : Java, MySQL

FRONT-END : Java

BACK-END : MySQL

OPERATING SYSTEM : Windows 10 **SAMPLE CODE**

3 SAMPLE CODE:

3.1.ADMIN DASHBOARD

```
public class AdminDashBoard extends javax.swing.JFrame {

    /**
* Creates new form AdminDashBoard
     */
    public AdminDashBoard() {
        initComponents();
    }
/**
* This method is called from within the constructor to initialize the form.      *
   WARNING: Do NOT modify this code. The content of this method is always
   * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
btnTicketBooking = new javax.swing.JButton();
btnTrainSchedule = new javax.swing.JButton();
btnTrainFind = new javax.swing.JButton();
btnFareEnquiry = new javax.swing.JButton();
```

```
btnAddTrain = new javax.swing.JButton();
btnPassengerDetails = new javax.swing.JButton();
        btnLogout = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 36)); // NOI18N
jLabel1.setForeground(new java.awt.Color(0, 0, 204));
        jLabel1.setText("ADMIN DASHBORD");

        btnTicketBooking.setBackground(new java.awt.Color(51, 255, 255));
btnTicketBooking.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnTicketBooking.setForeground(new java.awt.Color(153, 0, 153));
btnTicketBooking.setText("Ticket Booking");
btnTicketBooking.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
btnTicketBookingActionPerformed(evt);
            }
});

        btnTrainSchedule.setBackground(new java.awt.Color(51, 255, 255));
btnTrainSchedule.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnTrainSchedule.setForeground(new java.awt.Color(153, 0, 153));
btnTrainSchedule.setText("Train Schedule");
        btnTrainSchedule.addActionListener(new  java.awt.event.ActionListener()  {
public       void       actionPerformed(java.awt.event.ActionEvent       evt)       {
btnTrainScheduleActionPerformed(evt);
            }
});

        btnTrainFind.setBackground(new java.awt.Color(51, 255, 255));
btnTrainFind.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnTrainFind.setForeground(new java.awt.Color(153, 0, 153));
btnTrainFind.setText("Find Train");          btnTrainFind.addActionListener(new
java.awt.event.ActionListener() {              public void
actionPerformed(java.awt.event.ActionEvent evt) {
btnTrainFindActionPerformed(evt);
            }
});

        btnFareEnquiry.setBackground(new java.awt.Color(51, 255, 255));
btnFareEnquiry.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnFareEnquiry.setForeground(new java.awt.Color(153, 0, 153));
btnFareEnquiry.setText("Fare Enquiry");
btnFareEnquiry.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
btnFareEnquiryActionPerformed(evt);
            }
});

        btnAddTrain.setBackground(new java.awt.Color(51, 255, 255));
btnAddTrain.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
      btnAddTrain.setForeground(new java.awt.Color(153, 0, 153));
btnAddTrain.setText("Add Trains");
      btnAddTrain.addActionListener(new java.awt.event.ActionListener() {
       public void actionPerformed(java.awt.event.ActionEvent evt) {
btnAddTrainActionPerformed(evt);
            }
```

```
    });

    btnPassengerDetails.setBackground(new java.awt.Color(51, 255, 255));
btnPassengerDetails.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnPassengerDetails.setForeground(new java.awt.Color(153, 0, 153));
btnPassengerDetails.setText("Passenger Details");
    btnPassengerDetails.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
btnPassengerDetailsActionPerformed(evt);
        }
});

    btnLogout.setBackground(new java.awt.Color(51, 255, 255));
btnLogout.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnLogout.setForeground(new java.awt.Color(153, 0, 153));
btnLogout.setText("Logout");        btnLogout.addActionListener(new
java.awt.event.ActionListener() {            public void
actionPerformed(java.awt.event.ActionEvent evt) {
btnLogoutActionPerformed(evt);
        }
});

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);        layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap(141, Short.MAX_VALUE)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(btnAddTrain, javax.swing.GroupLayout.PREFERRED_SIZE, 202,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(btnTrainFind, javax.swing.GroupLayout.PREFERRED_SIZE, 202,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(btnTicketBooking, javax.swing.GroupLayout.PREFERRED_SIZE, 379,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGap(224, 224, 224)
                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
.addComponent(btnTrainSchedule, javax.swing.GroupLayout.PREFERRED_SIZE, 202,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(btnFareEnquiry, javax.swing.GroupLayout.PREFERRED_SIZE, 202,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(btnPassengerDetails, javax.swing.GroupLayout.PREFERRED_SIZE, 202,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGroup(layout.createSequentialGroup()
                    .addGap(329, 329, 329)
                    .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 467,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addGap(139, 139, 139))
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
.addGap(0, 0, Short.MAX_VALUE)
            .addComponent(btnLogout, javax.swing.GroupLayout.PREFERRED_SIZE, 224,
javax.swing.GroupLayout.PREFERRED_SIZE))
    );

    layout.linkSize(javax.swing.SwingConstants.HORIZONTAL, new java.awt.Component[] {btnAddTrain, btnFareEnquiry,
btnPassengerDetails, btnTicketBooking, btnTrainFind, btnTrainSchedule});
```

```
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(btnLogout)
                .addGap(27, 27, 27)
                .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 57,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(35, 35, 35)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnTicketBooking, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnTrainSchedule, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(69, 69, 69)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(btnTrainFind, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnFareEnquiry, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(82, 82, 82)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(btnAddTrain, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnPassengerDetails, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(183, Short.MAX_VALUE))
        );

        layout.linkSize(javax.swing.SwingConstants.VERTICAL, new java.awt.Component[] {btnAddTrain, btnFareEnquiry,
btnPassengerDetails, btnTicketBooking, btnTrainFind, btnTrainSchedule});

        pack();
    }// </editor-fold>//GEN-END:initComponents
    // Button for performing actions     private void
btnTicketBookingActionPerformed(java.awt.event.ActionEvent evt)
{//GENFIRST:event_btnTicketBookingActionPerformed
        TicketBookingForm tbf = new TicketBookingForm();
tbf.setVisible(true);
        this.dispose();

    }//GEN-LAST:event_btnTicketBookingActionPerformed

    private void btnTrainScheduleActionPerformed(java.awt.event.ActionEvent evt)
{//GENFIRST:event_btnTrainScheduleActionPerformed
        TrainScheduleForm ts = new TrainScheduleForm();
ts.setVisible(true);        this.dispose();
    }//GEN-LAST:event_btnTrainScheduleActionPerformed

    private void btnTrainFindActionPerformed(java.awt.event.ActionEvent evt)
{//GENFIRST:event_btnTrainFindActionPerformed
        TrainBetweenStation tbs = new TrainBetweenStation();
tbs.setVisible(true);        this.dispose();
    }//GEN-LAST:event_btnTrainFindActionPerformed
```

```
    private void btnFareEnquiryActionPerformed(java.awt.event.ActionEvent evt)
{//GENFIRST:event_btnFareEnquiryActionPerformed
    TrainFareEnquiryForm tfq = new TrainFareEnquiryForm();
tfq.setVisible(true);         this.dispose();
   }//GEN-LAST:event_btnFareEnquiryActionPerformed

   private void btnAddTrainActionPerformed(java.awt.event.ActionEvent evt)
{//GENFIRST:event_btnAddTrainActionPerformed          TrainsAdd ta = new
TrainsAdd();
        ta.setVisible(true);
this.dispose();
    }//GEN-LAST:event_btnAddTrainActionPerformed

    private void btnPassengerDetailsActionPerformed(java.awt.event.ActionEvent evt)
{//GENFIRST:event_btnPassengerDetailsActionPerformed
        PassengerDetailsForm pdf = new PassengerDetailsForm();
pdf.setVisible(true);         this.dispose();
    }//GEN-LAST:event_btnPassengerDetailsActionPerformed

    private void btnLogoutActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_btnLogoutActionPerformed
        HomeScreen hs = new HomeScreen();
hs.setVisible(true);         this.dispose();
    }//GEN-LAST:event_btnLogoutActionPerformed


    // Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton btnAddTrain;     private
javax.swing.JButton btnFareEnquiry;     private
javax.swing.JButton btnLogout;     private javax.swing.JButton
btnPassengerDetails;     private javax.swing.JButton
btnTicketBooking;     private javax.swing.JButton btnTrainFind;
private javax.swing.JButton btnTrainSchedule;     private
javax.swing.JLabel jLabel1;
    // End of variables declaration//GEN-END:variables }


// Import packages import
java.awt.HeadlessException; import
java.sql.Connection; import
java.sql.DriverManager; import
java.sql.ResultSet; import
java.sql.SQLException; import
java.sql.Statement;
import javax.swing.JOptionPane;

public class AdminLogin extends javax.swing.JFrame {

    /**
* Creates new form AdminLogin
     */
    public AdminLogin() {
        initComponents();
    }      /**      * This method is called from within the constructor to
initialize the form.
* WARNING: Do NOT modify this code. The content of this method is always
* regenerated by the Form Editor.     */
```

```
    @SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
private void initComponents() {

        jLabel1 = new javax.swing.JLabel();         jLabel2
= new javax.swing.JLabel();         jLabel3 = new
javax.swing.JLabel();         txtUserName = new
javax.swing.JTextField();         btnLogin = new
javax.swing.JButton();         btnCancel = new
javax.swing.JButton();         btnForgotPassword = new
javax.swing.JButton();         btnRegistration = new
javax.swing.JButton();         jScrollPane1 = new
javax.swing.JScrollPane();
        txtPasword = new javax.swing.JPasswordField();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 36)); // NOI18N
jLabel1.setForeground(new java.awt.Color(0, 0, 204));         jLabel1.setText("ADMIN
LOGIN");

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 16)); // NOI18N
jLabel2.setForeground(new java.awt.Color(204, 0, 51));         jLabel2.setText("User
Name");

        jLabel3.setFont(new java.awt.Font("Tahoma", 1, 16)); // NOI18N
jLabel3.setForeground(new java.awt.Color(204, 0, 51));         jLabel3.setText("Password");

        txtUserName.setForeground(new java.awt.Color(0, 153, 153));

        btnLogin.setBackground(new java.awt.Color(51, 255, 255));
btnLogin.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnLogin.setForeground(new java.awt.Color(153, 0, 153));
btnLogin.setText("Login");         btnLogin.addActionListener(new
java.awt.event.ActionListener() {             public void
actionPerformed(java.awt.event.ActionEvent evt) {
btnLoginActionPerformed(evt);
            }
});

        btnCancel.setBackground(new java.awt.Color(51, 255, 255));
btnCancel.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnCancel.setForeground(new java.awt.Color(153, 0, 153));
btnCancel.setText("Cancel");
        btnCancel.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
btnCancelActionPerformed(evt);
            }
});

       btnForgotPassword.setBackground(new java.awt.Color(51, 255, 255));
btnForgotPassword.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
btnForgotPassword.setForeground(new java.awt.Color(153, 0, 153));
btnForgotPassword.setText("Forget Password");
                btnForgotPassword.addActionListener(new java.awt.event.ActionListener() {
                    public void actionPerformed(java.awt.event.ActionEvent evt) {
btnForgotPasswordActionPerformed(evt);
```

```
        }
    });

       btnRegistration.setBackground(new java.awt.Color(51, 255, 255));
btnRegistration.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
btnRegistration.setForeground(new java.awt.Color(153, 0, 153));
        btnRegistration.setText("Registration");
btnRegistration.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
btnRegistrationActionPerformed(evt);
            }
});

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);          layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(530, 530, 530)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(378, Short.MAX_VALUE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 352,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 211,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtUserName, javax.swing.GroupLayout.PREFERRED_SIZE, 211,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 211,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(2, 2, 2)
                        .addComponent(txtPasword, javax.swing.GroupLayout.PREFERRED_SIZE, 210,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(btnLogin, javax.swing.GroupLayout.PREFERRED_SIZE, 133,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(39, 39, 39)
                        .addComponent(btnRegistration, javax.swing.GroupLayout.PREFERRED_SIZE, 150,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(50, 50, 50)
                        .addComponent(btnCancel, javax.swing.GroupLayout.PREFERRED_SIZE, 133,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(162, 162, 162)
                        .addComponent(btnForgotPassword, javax.swing.GroupLayout.PREFERRED_SIZE, 210,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(134, 134, 134))
        );
        layout.setVerticalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
            .addGap(40, 40, 40)
            .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 45,
javax.swing.GroupLayout.PREFERRED_SIZE)
          .addGap(61, 61, 61)
          .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(11, 11, 11)
            .addComponent(txtUserName, javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(53, 53, 53)
            .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(13, 13, 13)
            .addComponent(txtPasword, javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(80, 80, 80)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()
                .addGap(1, 1, 1)
                .addComponent(btnLogin, javax.swing.GroupLayout.PREFERRED_SIZE, 38,
javax.swing.GroupLayout.PREFERRED_SIZE))
              .addComponent(btnRegistration, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
javax.swing.GroupLayout.PREFERRED_SIZE)
              .addComponent(btnCancel, javax.swing.GroupLayout.PREFERRED_SIZE, 38,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(60, 60, 60)
            .addComponent(btnForgotPassword, javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(90, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents
```

3.2.DB CONNECTION

```
    import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class App {

    public static void main(String[] args) {
        // MySQL database credentials
        String url = "jdbc:mysql://localhost:3306/busbookingsystem"; // Change localhost
and database name
        String username = "root"; // Your MySQL username
        String password = "roots"; // Your MySQL password

        Connection connection = null;
        Statement statement = null;
        ResultSet resultSet = null;
```

```
        try {
            // Establish connection to the database
            connection = DriverManager.getConnection(url, username, password);
            System.out.println("Connection established successfully!");

            // Create a statement object
            statement = connection.createStatement();

            // Execute a query (e.g., fetching all rows from a table)
            String sql = "SELECT * FROM your_table_name"; // Replace with your actual
table
            resultSet = statement.executeQuery(sql);

            // Process the result set
            while (resultSet.next()) {
                System.out.println("ID: " + resultSet.getInt("id") + ", Name: " +
resultSet.getString("name"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                // Close the resources to avoid memory leaks
                if (resultSet != null) resultSet.close();
                if (statement != null) statement.close();
                if (connection != null) connection.close();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }
}
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class App {

    public static void main(String[] args) {
        // MySQL database credentials
        String url = "jdbc:mysql://localhost:3306/busbookingsystem"; // Change localhost
and database name
        String username = "root"; // Your MySQL username
        String password = "roots"; // Your MySQL password

        Connection connection = null;
        Statement statement = null;
        ResultSet resultSet = null;

        try {
            // Establish connection to the database
            connection = DriverManager.getConnection(url, username, password);
```

```
            System.out.println("Connection established successfully!");

            // Create a statement object
            statement = connection.createStatement();

            // Execute a query (e.g., fetching all rows from a table)
            String sql = "SELECT * FROM your_table_name"; // Replace with your actual
table
            resultSet = statement.executeQuery(sql);

            // Process the result set
            while (resultSet.next()) {
                System.out.println("ID: " + resultSet.getInt("id") + ", Name: " +
resultSet.getString("name"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                // Close the resources to avoid memory leaks
                if (resultSet != null) resultSet.close();
                if (statement != null) statement.close();
                if (connection != null) connection.close();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }
}
```

3.3.MAIN APP

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
import javax.swing.table.DefaultTableModel;

public class UserAuthApp {

    private static final String DB_URL = "jdbc:mysql://localhost:3306/busbookingsystem";
// Your MySQL DB URL
    private static final String DB_USERNAME = "root"; // Replace with your MySQL
username
    private static final String DB_PASSWORD = "roots"; // Replace with your MySQL
password

    // Method to open the Edit File page for adding passenger details
    private static void openEditFilePage() {
        // Create a new Edit File page frame
        JFrame editFrame = new JFrame("Add Passenger Details");
        editFrame.setSize(400, 300);
        editFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
        // Create input fields for passenger details
        JLabel nameLabel = new JLabel("Passenger Name:");
        JTextField nameField = new JTextField(20);

        JLabel numberLabel = new JLabel("Passenger Number:");
        JTextField numberField = new JTextField(20);

        JLabel destinationLabel = new JLabel("Destination:");
        JTextField destinationField = new JTextField(20);

        JLabel busNoLabel = new JLabel("Bus Number:");
        JTextField busNoField = new JTextField(20);

        // Save button to save the data to the database
        JButton saveButton = new JButton("Save");
        saveButton.setFont(new Font("Arial", Font.BOLD, 16));

        // Create a panel to organize the form fields and button
        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(5, 2, 10, 10));  // 5 rows and 2 columns
        panel.add(nameLabel);
        panel.add(nameField);
        panel.add(numberLabel);
        panel.add(numberField);
        panel.add(destinationLabel);
        panel.add(destinationField);
        panel.add(busNoLabel);
        panel.add(busNoField);
        panel.add(new JLabel());  // Empty cell to maintain grid structure
        panel.add(saveButton);

        // Add panel to the frame
        editFrame.add(panel, BorderLayout.CENTER);

        // Action listener for the Save button
        saveButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String name = nameField.getText();
                String number = numberField.getText();
                String destination = destinationField.getText();
                String busNumber = busNoField.getText();

                // Save the passenger details to the database
                if (name.isEmpty() || number.isEmpty() || destination.isEmpty() ||
busNumber.isEmpty()) {
                    JOptionPane.showMessageDialog(editFrame, "Please fill all the fields.");
                } else {
                    // Insert data into the database
                    boolean success = savePassengerToDatabase(name, number, destination,
busNumber);
                    if (success) {
                        JOptionPane.showMessageDialog(editFrame, "Passenger details saved
successfully.");
```

```
                editFrame.dispose();  // Close the "Edit File" window
            } else {
                JOptionPane.showMessageDialog(editFrame, "Error saving data.");
            }
        }
    }
});

    // Show the Edit File page
    editFrame.setVisible(true);
}

// Method to save passenger details to the database
private static boolean savePassengerToDatabase(String name, String number, String
destination, String busNumber) {
    String query = "INSERT INTO bus (name, number, destination, bus_number)
VALUES (?, ?, ?, ?)";

    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
         PreparedStatement stmt = conn.prepareStatement(query)) {

        stmt.setString(1, name);
        stmt.setString(2, number);
        stmt.setString(3, destination);
        stmt.setString(4, busNumber);

        int rowsInserted = stmt.executeUpdate();
        return rowsInserted > 0; // Return true if data was successfully inserted
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return false;  // Return false if there was an error
}

// Method to open the List Passenger page and display the data from the 'bus' table
private static void openListPage() {
    // Create a new frame for listing passengers
    JFrame listFrame = new JFrame("List of Passengers");
    listFrame.setSize(600, 400);
    listFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

    // Create a table model to hold the data from the database
    DefaultTableModel model = new DefaultTableModel();
    model.addColumn("ID");
    model.addColumn("Name");
    model.addColumn("Number");
    model.addColumn("Destination");
    model.addColumn("Bus Number");

    // Create the JTable to display the data
    JTable table = new JTable(model);
    JScrollPane scrollPane = new JScrollPane(table);
```

```
        // Create the search bar at the top
        JPanel searchPanel = new JPanel();
        searchPanel.setLayout(new FlowLayout(FlowLayout.LEFT));

        JLabel searchLabel = new JLabel("Search by ID or Name: ");
        JTextField searchField = new JTextField(20);
        JButton searchButton = new JButton("Search");

        searchPanel.add(searchLabel);
        searchPanel.add(searchField);
        searchPanel.add(searchButton);

        listFrame.add(searchPanel, BorderLayout.NORTH);
        listFrame.add(scrollPane, BorderLayout.CENTER);

        // Load the data from the database and populate the table
        loadDataIntoTable(model, table);

        // Create a delete button to remove the selected row
        JButton deleteButton = new JButton("Delete");
        deleteButton.setFont(new Font("Arial", Font.BOLD, 16));
        deleteButton.setPreferredSize(new Dimension(100, 40));

        deleteButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                int selectedRow = table.getSelectedRow(); // Get the selected row index
                if (selectedRow != -1) { // If a row is selected
                    int id = (int) model.getValueAt(selectedRow, 0); // Get the ID of the selected
row
                    boolean success = deletePassengerFromDatabase(id); // Delete from the
database
                    if (success) {
                        model.removeRow(selectedRow); // Remove the row from the JTable
                        JOptionPane.showMessageDialog(listFrame, "Passenger deleted
successfully.");
                    } else {
                        JOptionPane.showMessageDialog(listFrame, "Error deleting passenger.");
                    }
                } else {
                    JOptionPane.showMessageDialog(listFrame, "Please select a row to
delete.");
                }
            }
        });

        // Add the delete button to the bottom-right of the frame
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout(FlowLayout.RIGHT));
        buttonPanel.add(deleteButton);
        listFrame.add(buttonPanel, BorderLayout.SOUTH);

        // Add the search functionality
        searchButton.addActionListener(new ActionListener() {
```

```
            @Override
            public void actionPerformed(ActionEvent e) {
                String searchText = searchField.getText().toLowerCase().trim();

                if (!searchText.isEmpty()) {
                    searchPassengers(searchText, model);
                } else {
                    loadDataIntoTable(model, table); // Reload all data if the search field is
empty
                }
            }
        });

        // Show the List Passenger page
        listFrame.setVisible(true);
    }

    // Method to load data from the 'bus' table into the JTable
    private static void loadDataIntoTable(DefaultTableModel model, JTable table) {
        model.setRowCount(0); // Clear existing rows

        String query = "SELECT * FROM bus";

        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            // Loop through the result set and add rows to the table model
            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                String number = rs.getString("number");
                String destination = rs.getString("destination");
                String busNumber = rs.getString("bus_number");

                model.addRow(new Object[]{id, name, number, destination, busNumber});
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Method to search passengers by ID or Name
    private static void searchPassengers(String searchText, DefaultTableModel model) {
        model.setRowCount(0); // Clear existing rows

        String query = "SELECT * FROM bus WHERE id LIKE ? OR name LIKE ?";

        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
             PreparedStatement stmt = conn.prepareStatement(query)) {

            stmt.setString(1, "%" + searchText + "%");
```

```
            stmt.setString(2, "%" + searchText + "%");

            ResultSet rs = stmt.executeQuery();

            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                String number = rs.getString("number");
                String destination = rs.getString("destination");
                String busNumber = rs.getString("bus_number");

                model.addRow(new Object[]{id, name, number, destination, busNumber});
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Method to delete a passenger from the database
    private static boolean deletePassengerFromDatabase(int id) {
        String query = "DELETE FROM bus WHERE id = ?";

        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
             PreparedStatement stmt = conn.prepareStatement(query)) {

            stmt.setInt(1, id);

            int rowsDeleted = stmt.executeUpdate();
            return rowsDeleted > 0; // Return true if a row was deleted
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return false;  // Return false if there was an error
    }

    public static void main(String[] args) {
        // Example entry point for the application
        JFrame frame = new JFrame("Bus Booking System");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);

        // Create a button to open the Add Passenger form
        JButton openEditFileButton = new JButton("Add Passenger");
        openEditFileButton.setFont(new Font("Arial", Font.BOLD, 16));
        openEditFileButton.setPreferredSize(new Dimension(200, 40));

        openEditFileButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                openEditFilePage();  // Open the Add Passenger form when clicked
            }
        });
```

```
        // Create a button to list all passengers
        JButton listPassengersButton = new JButton("List Passengers");
        listPassengersButton.setFont(new Font("Arial", Font.BOLD, 16));
        listPassengersButton.setPreferredSize(new Dimension(200, 40));

        listPassengersButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                openListPage();  // Open the List Passengers page when clicked
            }
        });

        // Set layout and add buttons to the center of the frame using
BorderLayout.CENTER
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 20, 10));
        buttonPanel.add(openEditFileButton);
        buttonPanel.add(listPassengersButton);

        frame.setLayout(new BorderLayout());
        frame.add(buttonPanel, BorderLayout.CENTER);  // Add the button panel at the
center

        frame.setVisible(true);
    }
}
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
import javax.swing.table.DefaultTableModel;

public class UserAuthApp {

    private static final String DB_URL = "jdbc:mysql://localhost:3306/busbookingsystem";
// Your MySQL DB URL
    private static final String DB_USERNAME = "root"; // Replace with your MySQL
username
    private static final String DB_PASSWORD = "roots"; // Replace with your MySQL
password

    // Method to open the Edit File page for adding passenger details
    private static void openEditFilePage() {
        // Create a new Edit File page frame
        JFrame editFrame = new JFrame("Add Passenger Details");
        editFrame.setSize(400, 300);
        editFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

        // Create input fields for passenger details
        JLabel nameLabel = new JLabel("Passenger Name:");
        JTextField nameField = new JTextField(20);

        JLabel numberLabel = new JLabel("Passenger Number:");
```

```
    JTextField numberField = new JTextField(20);

    JLabel destinationLabel = new JLabel("Destination:");
    JTextField destinationField = new JTextField(20);

    JLabel busNoLabel = new JLabel("Bus Number:");
    JTextField busNoField = new JTextField(20);

    // Save button to save the data to the database
    JButton saveButton = new JButton("Save");
    saveButton.setFont(new Font("Arial", Font.BOLD, 16));

    // Create a panel to organize the form fields and button
    JPanel panel = new JPanel();
    panel.setLayout(new GridLayout(5, 2, 10, 10));  // 5 rows and 2 columns
    panel.add(nameLabel);
    panel.add(nameField);
    panel.add(numberLabel);
    panel.add(numberField);
    panel.add(destinationLabel);
    panel.add(destinationField);
    panel.add(busNoLabel);
    panel.add(busNoField);
    panel.add(new JLabel());  // Empty cell to maintain grid structure
    panel.add(saveButton);

    // Add panel to the frame
    editFrame.add(panel, BorderLayout.CENTER);

    // Action listener for the Save button
    saveButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            String name = nameField.getText();
            String number = numberField.getText();
            String destination = destinationField.getText();
            String busNumber = busNoField.getText();

            // Save the passenger details to the database
            if (name.isEmpty() || number.isEmpty() || destination.isEmpty() ||
busNumber.isEmpty()) {
                JOptionPane.showMessageDialog(editFrame, "Please fill all the fields.");
            } else {
                // Insert data into the database
                boolean success = savePassengerToDatabase(name, number, destination,
busNumber);
                if (success) {
                    JOptionPane.showMessageDialog(editFrame, "Passenger details saved
successfully.");
                    editFrame.dispose();  // Close the "Edit File" window
                } else {
                    JOptionPane.showMessageDialog(editFrame, "Error saving data.");
                }
            }
```

```
        }
    });

    // Show the Edit File page
    editFrame.setVisible(true);
}

// Method to save passenger details to the database
private static boolean savePassengerToDatabase(String name, String number, String
destination, String busNumber) {
    String query = "INSERT INTO bus (name, number, destination, bus_number)
VALUES (?, ?, ?, ?)";

    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
         PreparedStatement stmt = conn.prepareStatement(query)) {

        stmt.setString(1, name);
        stmt.setString(2, number);
        stmt.setString(3, destination);
        stmt.setString(4, busNumber);

        int rowsInserted = stmt.executeUpdate();
        return rowsInserted > 0; // Return true if data was successfully inserted
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return false;  // Return false if there was an error
}

// Method to open the List Passenger page and display the data from the 'bus' table
private static void openListPage() {
    // Create a new frame for listing passengers
    JFrame listFrame = new JFrame("List of Passengers");
    listFrame.setSize(600, 400);
    listFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

    // Create a table model to hold the data from the database
    DefaultTableModel model = new DefaultTableModel();
    model.addColumn("ID");
    model.addColumn("Name");
    model.addColumn("Number");
    model.addColumn("Destination");
    model.addColumn("Bus Number");

    // Create the JTable to display the data
    JTable table = new JTable(model);
    JScrollPane scrollPane = new JScrollPane(table);

    // Create the search bar at the top
    JPanel searchPanel = new JPanel();
    searchPanel.setLayout(new FlowLayout(FlowLayout.LEFT));

    JLabel searchLabel = new JLabel("Search by ID or Name: ");
```

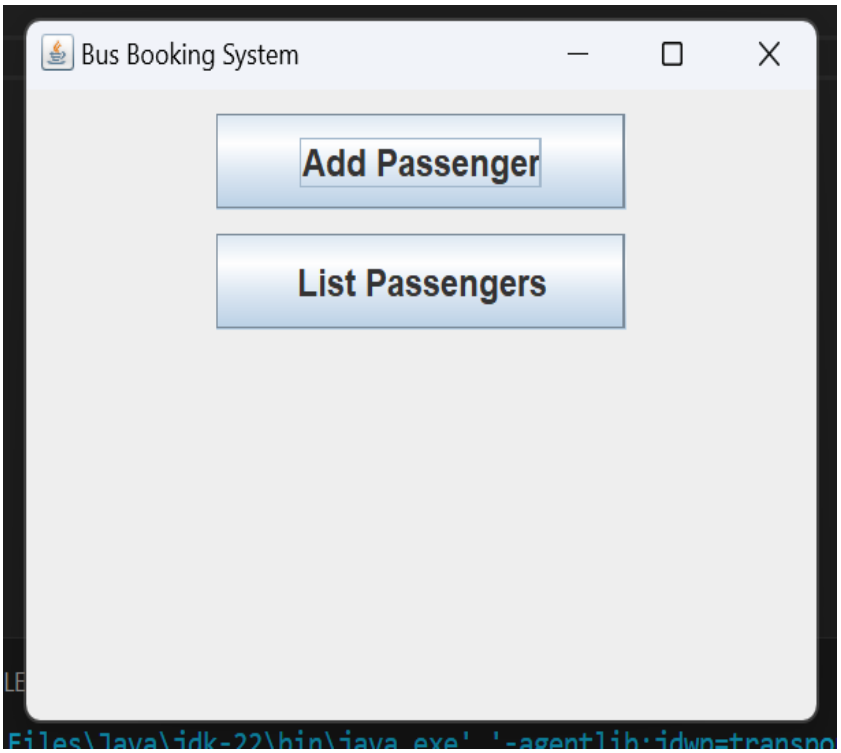
```
        JTextField searchField = new JTextField(20);
        JButton searchButton = new JButton("Search");

        searchPanel.add(searchLabel);
        searchPanel.add(searchField);
        searchPanel.add(searchButton);

        listFrame.add(searchPanel, BorderLayout.NORTH);
        listFrame.add(scrollPane, BorderLayout.CENTER);

        // Load the data from the database and populate the table
        loadDataIntoTable(model, table);

        // Create a delete button to remove the selected row
        JButton deleteButton = new JButton("Delete");
        deleteButton.setFont(new Font("Arial", Font.BOLD, 16));
        deleteButton.setPreferredSize(new Dimension(100, 40));

        deleteButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                int selectedRow = table.getSelectedRow(); // Get the selected row index
                if (selectedRow != -1) { // If a row is selected
                    int id = (int) model.getValueAt(selectedRow, 0); // Get the ID of the selected
row
                    boolean success = deletePassengerFromDatabase(id); // Delete from the
database
                    if (success) {
                        model.removeRow(selectedRow); // Remove the row from the JTable
                        JOptionPane.showMessageDialog(listFrame, "Passenger deleted
successfully.");
                    } else {
                        JOptionPane.showMessageDialog(listFrame, "Error deleting passenger.");
                    }
                } else {
                    JOptionPane.showMessageDialog(listFrame, "Please select a row to
delete.");
                }
            }
        });

        // Add the delete button to the bottom-right of the frame
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout(FlowLayout.RIGHT));
        buttonPanel.add(deleteButton);
        listFrame.add(buttonPanel, BorderLayout.SOUTH);

        // Add the search functionality
        searchButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String searchText = searchField.getText().toLowerCase().trim();

                if (!searchText.isEmpty()) {
```

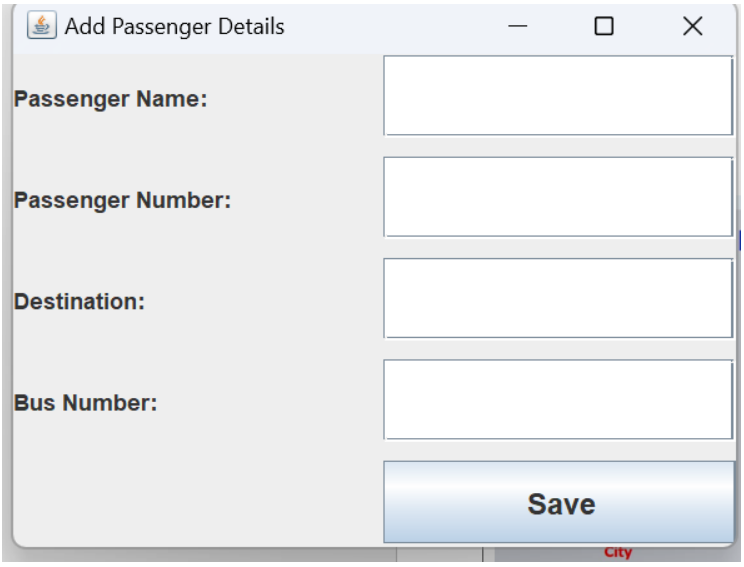
```
                searchPassengers(searchText, model);
            } else {
                loadDataIntoTable(model, table); // Reload all data if the search field is
empty
            }
        }
    });

    // Show the List Passenger page
    listFrame.setVisible(true);
  }

  // Method to load data from the 'bus' table into the JTable
  private static void loadDataIntoTable(DefaultTableModel model, JTable table) {
    model.setRowCount(0); // Clear existing rows

    String query = "SELECT * FROM bus";

    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery(query)) {

        // Loop through the result set and add rows to the table model
        while (rs.next()) {
            int id = rs.getInt("id");
            String name = rs.getString("name");
            String number = rs.getString("number");
            String destination = rs.getString("destination");
            String busNumber = rs.getString("bus_number");

            model.addRow(new Object[]{id, name, number, destination, busNumber});
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }

  // Method to search passengers by ID or Name
  private static void searchPassengers(String searchText, DefaultTableModel model) {
    model.setRowCount(0); // Clear existing rows

    String query = "SELECT * FROM bus WHERE id LIKE ? OR name LIKE ?";

    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
         PreparedStatement stmt = conn.prepareStatement(query)) {

        stmt.setString(1, "%" + searchText + "%");
        stmt.setString(2, "%" + searchText + "%");

        ResultSet rs = stmt.executeQuery();

        while (rs.next()) {
```

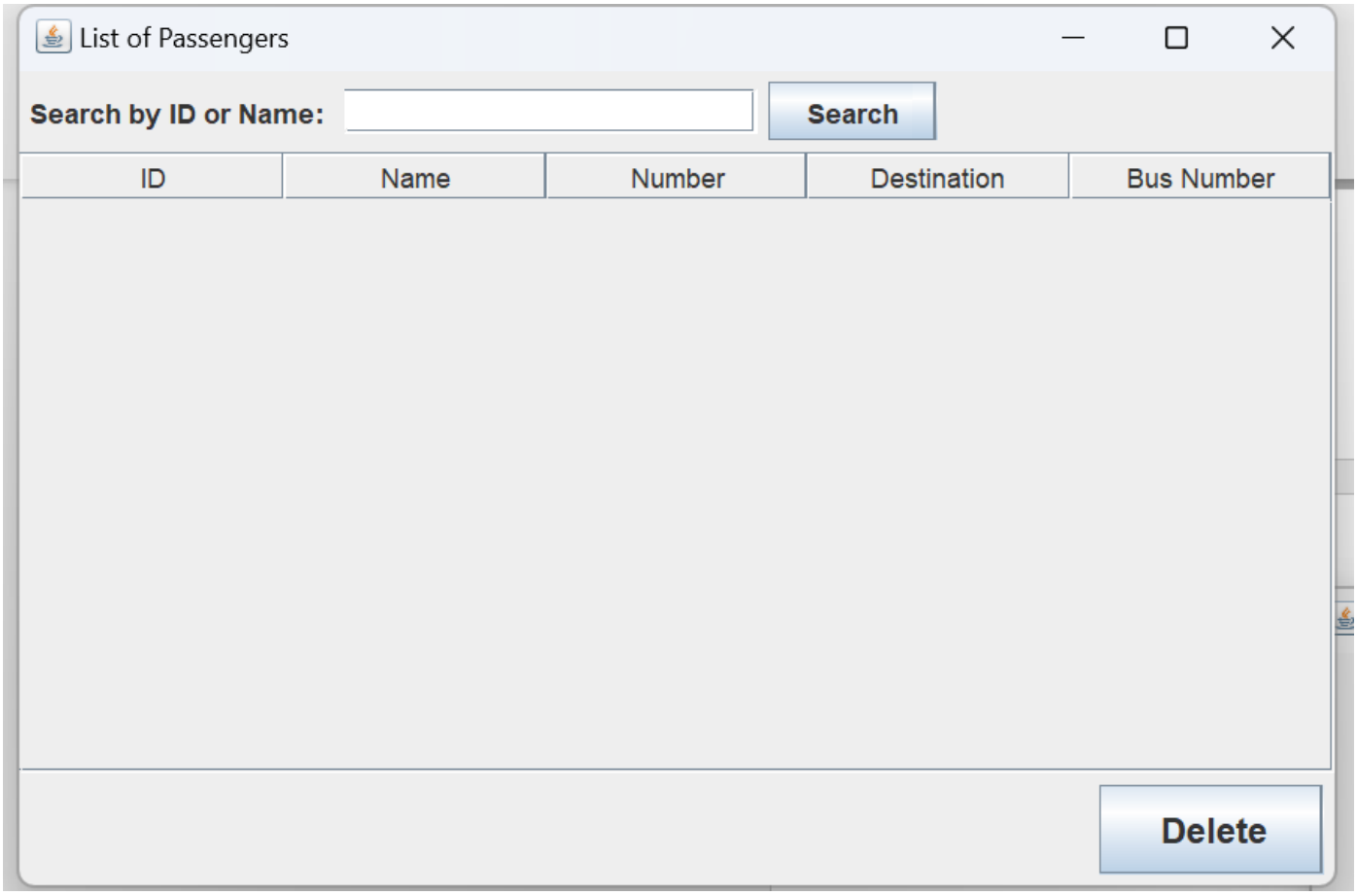
```
            int id = rs.getInt("id");
            String name = rs.getString("name");
            String number = rs.getString("number");
            String destination = rs.getString("destination");
            String busNumber = rs.getString("bus_number");

            model.addRow(new Object[]{id, name, number, destination, busNumber});
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Method to delete a passenger from the database
private static boolean deletePassengerFromDatabase(int id) {
    String query = "DELETE FROM bus WHERE id = ?";

    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USERNAME,
DB_PASSWORD);
         PreparedStatement stmt = conn.prepareStatement(query)) {

        stmt.setInt(1, id);

        int rowsDeleted = stmt.executeUpdate();
        return rowsDeleted > 0; // Return true if a row was deleted
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return false;  // Return false if there was an error
}

public static void main(String[] args) {
    // Example entry point for the application
    JFrame frame = new JFrame("Bus Booking System");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 300);

    // Create a button to open the Add Passenger form
    JButton openEditFileButton = new JButton("Add Passenger");
    openEditFileButton.setFont(new Font("Arial", Font.BOLD, 16));
    openEditFileButton.setPreferredSize(new Dimension(200, 40));

    openEditFileButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            openEditFilePage();  // Open the Add Passenger form when clicked
        }
    });

    // Create a button to list all passengers
    JButton listPassengersButton = new JButton("List Passengers");
    listPassengersButton.setFont(new Font("Arial", Font.BOLD, 16));
    listPassengersButton.setPreferredSize(new Dimension(200, 40));
```

```
        listPassengersButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                openListPage();  // Open the List Passengers page when clicked
            }
        });

        // Set layout and add buttons to the center of the frame using
BorderLayout.CENTER
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 20, 10));
        buttonPanel.add(openEditFileButton);
        buttonPanel.add(listPassengersButton);

        frame.setLayout(new BorderLayout());
        frame.add(buttonPanel, BorderLayout.CENTER);  // Add the button panel at the
center

        frame.setVisible(true);
    }
}
```

SNAPSHOTS:


Bus Booking System
Add Passenger
List Passengers


Add Passenger Details
Passenger Name:
Passenger Number:
Destination:
Bus Number:
Save


List of Passengers
Search by ID or Name:
Search
ID
Name
Number
Destination
Bus Number
Delete

CONCLUSION:

In conclusion, the Railway Ticket Booking and Management System successfully addresses the complexities associated with train ticket reservations and management. By utilizing Java for developing the front-end and MySQL for back-end database management, the project ensures a highly responsive, secure, and user-friendly system. The implementation of features such as real-time seat availability, ticket booking, cancellation, and administrative management simplifies the entire process for both passengers and railway operators.

This system not only automates the ticketing process but also contributes to better utilization of resources, reduced errors, and a more transparent and efficient railway management system. As a final product, it showcases how modern technologies like Java and MySQL can be used to improve traditional transportation systems, ultimately leading to a better user experience and operational efficiency.

The project can be further enhanced by incorporating additional features such as dynamic pricing, realtime train tracking, and mobile app integration to provide an even more comprehensive solution for railway ticket booking and management.

REFERENCES:

1. https://www.javatpoint.com/java-tutorial
2. https://www.wikipedia.org/
3. https://www.w3schools.com/sql/
4. SQL | Codecademy